World-State Transformations for Neuro-symbolic Interactive Storytelling

Santiago Góngora[1] † **Luis Chiruzzo**[1] **Gonzalo Méndez**[2] **Pablo Gervás**[2]

[1] Instituto de Computación, Facultad de Ingeniería, Universidad de la República, Uruguay
[2] Facultad de Informática, Universidad Complutense de Madrid, Spain
† sgongora@fing.edu.uy

Abstract

Large Language Models (LLMs) have changed the possibilities of Interactive Storytelling systems that process free-text user input. However, as more of these systems are built, evidence continues to mount regarding the story coherence problems that arise when relying solely on them. Recent research suggests that LLMs can effectively predict state changes within rule-based Interactive Storytelling systems, triggering pre-programmed **world-state transformations**. In this paper, we conduct an exploratory evaluation of whether such *transformations* can serve as a catalyst for player expression while aiming to address the incoherence issues typical of purely LLM-based approaches. Building upon PAYADOR, a neuro-symbolic architecture, we conducted experiments using an open-source model (Llama 3 70B) and a closed-source model (Gemini 1.5 Flash), with testing in both English and Spanish. Eight participants played two scenarios, designed to assess different evaluation objectives. Our observations suggest *transformations* improve LLM error traceability and enable logic checks, thus maintaining world-state consistency, while encouraging players to interact with the environment through their unique, creative inputs.

Introduction

Interactive Storytelling (IS) is a broad term for any system that allows users to influence how a story develops (Trichopoulos, Alexandridis, and Caridakis 2023). These systems face a constant struggle to balance *player agency* (i.e., the level of control over the story the user has) with a coherent plot, as giving the user more control often compromises the story's consistency (Riedl and Bulitko 2012)

One of the most prominent types of IS systems is Interactive Fiction (IF), in which players engage with a simulated environment through text-based *player input* (or *user input*). These games were remarkably sophisticated for their time, addressing world-modeling through complex rule sets grounded in formal linguistic structures (Reed 2023). Limited by their parsers, fixed action sets and a lack of common sense understanding, they often stifled player creativity and expression, constraining player agency. Large Language Models (LLMs) could address these challenges and revolutionize Human-Machine co-creation (Ferreira 2026), yet they still struggle with narrative pace and logical consistency (Góngora et al. 2023; Tian et al. 2024; Teleki et al. 2025). These limitations highlight a critical need for methods that ground LLM generation in more stable structures.

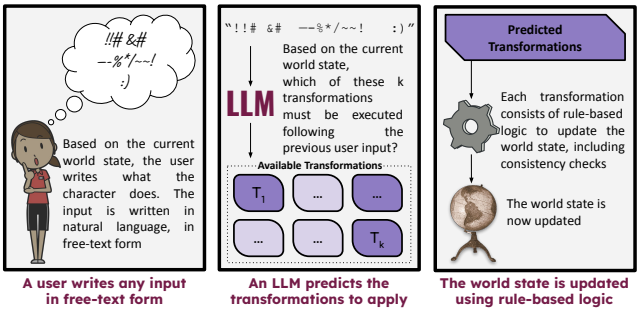

Figure 1: An overview of how *transformations* can be used to tackle the world-update problem.

In this paper, we present a pilot study of world-state *transformations*, a generalization of classic IF actions that are suggested by LLMs and symbolically executed in a previously defined structured world state (see Figure 1). We aim to evaluate whether such *transformations* provide a framework for IS systems designed to model a coherent world and allow players to express themselves through their own creativity. We first conceptually discuss *transformations*, and then detail the methodological design of our pilot study and the analysis of the game logs. Finally, we present the key observations and findings from the evaluation. The source code is available on GitHub[1].

Previous Work on the World-update Problem

The main topic of this paper is the **world-update** problem in IF systems, and text-based Interactive Storytelling in general. Although multiple definitions exist (Riedl and Bulitko 2012; Martin, Harrison, and Riedl 2016; Hausknecht et al. 2020; Callison-Burch et al. 2022; Góngora et al. 2024), the problem may be defined at a high level as follows:

$$action\ description\ (world\ state_n) \longrightarrow world\ state_{n+1}$$

where $n \in \mathbb{N}$ represents a point in time of the represented fictional world. That is, $world\ state_{n+1}$ is a consequence

[1] https://github.com/sgongora27/transformations

of doing the described $action$ in the $world\ state_n$[2].

Besides the parsers used in classic IF games like Zork—triggering a world-update process that was specifically pre-programmed for each available action (Reed 2023)—many recent works (e.g., AI Dungeon[3]) tried using LLMs for this purpose (Triyason 2023; You et al. 2024). The aforementioned cases are representative for two very different strategies to tackle this problem, but there are many other ways to do it. Xie et al. (2025) represent the *world state* through simple natural language sentences and use two fine-tuned LLMs to process the user input: one to detect invalid actions and another to generate outcomes for valid ones. Neuro-symbolic approaches (i.e. those combining artificial neural networks, like LLMs, and symbolic approaches) have also been explored to tackle this problem. Callison-Burch et al. (2022) fine-tuned a language model that tracks the game state and generates a response based on it. One of their main conclusions is that additional machinery would be needed to track the game state beyond a Language Model.

LLMs Predicting a Change in the World State

Within a brief period, some works have proposed using LLMs to update structured world-state representations from free-text user input.

PAYADOR (Góngora et al. 2024) is an approach to the world-update problem that tries to break with the tradition of using specific pre-programmed actions like in classic IF games, and focuses on modeling the general outcome of those actions instead. This conceptual shift draws directly from modeling Tabletop Role-playing Games (TTRPG) and how Game Masters (GM) reason about the world state in response to player actions: rather than relying on an objective methodology for every possible action (something impossible given the open-ended nature of these games) they evaluate how a player's intent within its unique context will impact the fictional world (Tychsen et al. 2005).

While PAYADOR was designed as a system-agnostic GM model, shortly thereafter, Song, Zhu, and Callison-Burch (2024) introduced a specialized GM model for "Jim Henson's Labyrinth: The Adventure Game". In this model, an LLM maintains an ad-hoc structured representation through function calls. In a contemporaneous effort, Wang et al. (2024) also addressed LLM reasoning over world-state updates, but focusing on realistic environments. While trying to check if LLMs can act as a reliable world model, they found sometimes this strategy can work better than predicting the whole world state each time.

World-state Transformations

The previously described works lead us to define what we call **world-state transformations**. *Transformations* are pre-programmed routines designed to symbolically update a world-state. While retaining the spirit of classic IF, they aim to funnel a wide range of player actions, including the unpredictable ones born from creative expression. For instance, a player might use a standard medkit or spell[4], or they could improvise by creating or bartering for a remedy: an interaction too context-dependent for hard-coded logic to anticipate. Such cases demand flexibility from the resulting IS system. *Transformations* may provide it, allowing LLMs to decide which of the available ones apply after any player input. Moreover, a single input may trigger one or more *transformations*, which collaborate together to shape the fictional world through synergic changes.

Classic approaches to the world-update problem triggered changes based on specific actions, limiting the player's options to a fixed set of outcomes predefined by the designers. Instead, *transformations* focus on general changes in the world state, regardless of the specific actions that trigger them. As a middle ground between purely neural systems and classic approaches, this shift in focus enables pre-programmed updates to the world state while allowing players to express themselves through their own vocabulary and unique ideas. As we will later discuss, *transformations* seem to have enabled our evaluators to follow this direction.

Methodology

As previously noted, both Góngora et al. (2024) and Song, Zhu, and Callison-Burch (2024) utilized world-state transformations within their respective IS systems. However, while the former did not include an empirical evaluation, the latter focused exclusively on human analysis of LLM-simulated sessions. Consequently, our objective was to conduct a pilot study with humans using a live system (allowing for real-time interaction) to address three research questions regarding *transformations*:

- Are LLMs capable of correctly suggesting *transformations* when grounded on a structured world state?
- Are *transformations* suitable to tackle the world-update problem?
- Do *transformations* encourage more creative player inputs?

Based on the PAYADOR source code[5], we implemented a system, designed two scenarios, and conducted an evaluation with eight players. We split the group evenly to cover two languages (English and Spanish) and two models (Gemini 1.5 Flash and Llama 3 70B).

Figure 1 illustrates how LLM-suggested *transformations* are used in our rule-based IS system. First, the symbolic IS system is built using an ad hoc representation that includes pre-programmed *transformations*, each having additional pre-programmed consistency checks. Then, every turn, an LLM receives the symbolic $world\ state$ (or a textual rendering of it), the free-text $player\ input$, and the list of available *transformations*, along with instructions on how to suggest them. After the LLM determines which *transformations* are triggered by the $player\ input$, the symbolic world model executes them subject to the corresponding consistency checks.

[2]Broadly speaking, it may happen that the described action has no impact in $world\ state_n$, in which case $world\ state_{n+1} = world\ state_n$.

[3]https://aidungeon.com/

[4]Classic IF actions a game designer typically considers.

[5]https://github.com/pln-fing-udelar/payador

System Description

The original PAYADOR version (Góngora et al. 2024) models the fictional world by considering `Items`, `Locations`, and `Characters`. We improved it by integrating `Puzzles` (see Table 1 in the appendices for details) and `Objectives`. Each scenario can have one of the following objectives: the player is in a `Location` or has an `Item`; the player is in the same `Location` of another `Character`; or an `Item` is in a specific `Location`. This was essential to conduct our experiments, as `Objectives` allow to establish clear completion criteria for our test scenarios, hence helping us determine if *transformations* alone can act as a vehicle in human-machine communication.

We started from the *transformations* included in the original PAYADOR version and improved them with robust consistency checks:

- *Moved Items* (MI-t): The LLM can suggest that an `Object` has to be moved from the player's inventory to the current `Location`, or vice-versa. It also includes the case of the player giving an NPC an `Object`, or vice-versa.
- *Unblocked Locations (UL-t)*: The LLM can suggest that a blocked `Location` is now reachable.
- *Player Movement (PM-t)*: The LLM can suggest that the player's `Character` has to be moved to a new `Location`.

Naturally, we could have included other *transformations*. However, to ensure a controlled experimental setting, we maintained this minimal set that aligns with traditional IF mechanics. Given that we placed no constraints on how evaluators wrote their inputs, this decision narrows the range of possibilities during the experimental phase.

Experimental Evaluation

In this section we will present the two pre-designed scenarios and the results of each playthrough[6]. We manually examined every turn of each playthrough, tagging them according to two type of errors: **LLM** and **World Modeling** errors.

The first type is related to errors found when the **LLM** predicts the *transformations* to be done: when the suggested *transformations* do not correctly represent the outcomes of the *user input*, or the *transformations* are not correctly written by the LLM. This error is further split into three categories, one for each *transformation* (*MI-t*, *UL-t* and *PM-t*). In order to provide further insight into these cases, the world-update prompt and examples for text-based world rendering are included in the appendices.

The second type involves **world modeling** errors: cases where the LLM correctly proposes *transformations*, but the final result fails to capture the player's intended actions or logical sequence due to system limitations. This time we break this type of error into two categories. The first, *Planning*, refers to the execution order of *transformations*. The system uses a default *MI-t* $\rightarrow$ *UL-t* $\rightarrow$ *PM-t* sequence, so it fails when a player needs to do things in a different order. The second one, *memory*, refers to the lack of memory in PAYADOR, where the only information stored is the one in the structured representation, as the contents of the prompt are reset every turn. This means that the rest of details, such as previous dialogue utterances, are not saved. Some errors arise when the player refers to details that were not stored.

[6]All the game logs are available on GitHub: https://github.com/sgongora27/transformations

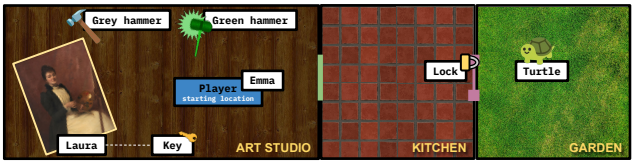

Figure 2: A graphical representation of Scenario A, including 2 `characters` (*Laura* and *Emma*), 5 `items` (*Grey hammer*, *Green hammer*, *Key*, *Lock* and *Turtle*), and 3 `locations` (*Art Studio*, *Kitchen* and *Garden*).

Scenario A

The first scenario consists of three *locations*. The player acts as *Emma*, a girl that has to find her *Turtle* and take it to the *Kitchen*. Starting in her mother's *Art Studio*, *Emma* has to get the *grey hammer* or the *key* (from her mother, *Laura*) to unlock the *Garden* door, where the *Turtle* is located. This sequence requires frequent world-state updates using all three *transformations*, aiming to intensely test them within a simple, realistic environment. Figure 2 illustrates the scenario; see the appendices for an enlarged version (Figure 3) and the manual analysis results in Table 2.

Overall, the eight testers managed to end the scenario. Some of them tried opening the lock with the *key*, and some of them broke it using the *hammer*. In both languages, errors in the LLM's *MI-t* suggestions were all traceable to an interesting issue: many players called the *Turtle Hojita* since the world state specifies that "Emma calls it *Hojita*" (e.g., "I place Hojita in the kitchen, and walk away to the garden again, to enjoy the rest of this sunny day"-TesterD). While we expected the LLM to resolve this co-reference, it occasionally suggested an incorrect *MI-t* using the name "Hojita", what was not recognized as a valid object by the world-modeling module, hence rejecting the *transformation*. Additionally, some *PM-t* errors occurred because the player input was not explicit (e.g., "As I walk through the door I look around trying to find my turtle, what do i see"-TesterC). A single *UL-t* error occurred when the LLM tried to unblock a location already shown as unblocked in its context.

Regarding world-modeling, many errors arose from the system's lack of *planning*. Players often tried to move to the *Kitchen* and drop the pet: an action requiring *PM-t* before *MI-t*. Our fixed execution order cannot accommodate this sequence, highlighting a limitation of the current default order. Finally, there was a single error due to the memory of the system, when TesterE said "agarro el otro martillo y me dirijo a la cocina" (*I take the other hammer and head to the kitchen*), but the LLM could not disambiguate between the *Grey hammer* and the *Green hammer*.

Scenario B

The second scenario also consists of three *locations*. This time, the player acts as *Venancio*, a *gaucho*[7] that has to find *Artigas*[8], who is locked in a *Cell* protected by a *Puzzle*. In order to enter the *Cell*, the player must first put out a *firewall*[9] and then solve a riddle. To put out the *firewall*, the player has to use *Venancio*'s magic to summon a giant wave of water, a superpower specified in the world state. In this case, the evaluation objective was twofold: first, to determine if LLM-predicted *transformations* adhere to non-realistic magical physics; second, to assess the LLM's ability to describe critical scene elements, such as *Venancio* summoning a water wave, ensuring players receive essential information to complete the scenario. In the appendices, Figure 4 illustrates this scenario, while Table 3 details its corresponding results.

As with Scenario A, all testers successfully completed the task. Notably, world-modeling errors dropped to zero as a result of limited environmental interaction. This trend extended to the *MI-t* category, where the sole error involved a consistency check rejecting TesterE's attempt to improvise an undeclared `Item`. However, this scenario yielded several notable *UL-t* errors. First, the particularity of having to solve a `Puzzle` to open the *Cell* door introduced interesting cases. Despite having the correct answer in-context, the LLM incorrectly validated TesterA and TesterC's wrong riddle answers, suggesting a *UL-t*, hence unblocking the *Cell*. These cases illustrate how symbolic grounding mitigates LLM-dependent errors, but cannot entirely eliminate them. Second, during TesterF's session, the LLM suggested an *UL-t* for an already reachable place, mirroring TesterG's experience in Scenario A. Finally, while the wall of fire challenge sparked unconventional (and successful) ideas among the players, the LLM did not suggest a *UL-t* when TesterB attempted to summon the wave by playing Venancio's guitar.

Transformations and Co-Creativity

Having discussed our observations from the qualitative analysis of errors, we will now explore the implications of *transformations* for Interactive Storytelling systems.

We will start from the most salient fact: all players were able to complete the scenarios. Our results indicate no substantial disparity in performance between the open-source model (Llama) and the closed-source one (Gemini), remaining consistent across both English and Spanish. Although our experiments are preliminary, the results suggest that *transformations* effectively bridge human-machine communication in LLM-enhanced IS systems, and that current LLMs are already capable of handling them.

Moreover, we observe that *transformations* seem to foster player creativity. For instance, TesterE unexpectedly used a window mentioned in the world state to enter the *Garden*, bypassing the intended *Lock*; and TesterG tried to summon the wave of water by playing "the call of the water" on the *guitar*. More generally, players in both scenarios engaged by role-playing their characters and adding creative details to their inputs (e.g., speaking with *Laura* in Scenario A, examining the magical setting or attempting to shout in the *Silent Zone* in Scenario B). This capability to leverage creative inputs may prove useful for systems requiring varying degrees of improvisation in human-machine co-creation. Consequently, *transformations* may be suitable for modeling TTRPGs, which scholars characterize as a form of organized, collaborative storytelling driven by rich player-GM interactions (Mäyrä 2017; Katifori et al. 2022).

While our experiments are preliminary, their results suggest that *transformations* offer a promising path for consistent world updates in IS systems. Evidence indicates they broaden player expression by providing numerous options to trigger pre-programmed updates, making them ideal for improvisational systems like TTRPG models. Although LLM errors may impact world coherence, the symbolic nature of *transformations* facilitates the integration of logic to verify and refine outputs. Following the idea that improvisational storytelling is limited only by player imagination (Martin, Harrison, and Riedl 2016), our results suggest that *transformations* represent a step forward in this direction.

Conclusions

In this paper, we evaluated world-state transformations as a method for tackling the *world-update* problem in Interactive Storytelling. By using an LLM to suggest pre-programmed world-state *transformations*, the system moves away from manually defined actions toward **a model of general world changes**. This shift in focus encourages players to interact more creatively with the environment while maintaining the controllability inherent in symbolic world-state representations.

Our pilot study with eight participants across two scenarios confirmed the viability of this approach: all players successfully completed both scenarios and engaged in co-creation while acting within the fictional worlds. While the system inherits common LLM limitations, leading to errors like prematurely solved puzzles, the symbolic nature of *transformations* allows for consistency checks that improve traceability and provide more transparent explainability than purely neural approaches.

Our experimental findings suggest several directions for further research. First, future work should explore creating world components from scratch, extending our recent preliminary efforts (Vaucher et al. 2026), and combining *transformations* to create them on the fly, thereby better modeling the improvisation inherent in TTRPGs. As the number of *transformations* grows, incorporating a planner to dynamically order their execution will become essential. Finally, to minimize the impact of incorrect LLM outputs, we aim to optimize how consistency checks validate them.

We hope this work inspires other researchers to push the boundaries of human-machine co-creation in Interactive Storytelling, and the application of TTRPGs models to different domains.

[7] According to the Cambridge Dictionary: *a South American cowboy who is skilled at riding a horse*.

[8] A historical figure of South America.

[9] After running all the experiments we realized this component should be called *Wall of Fire*. Apparently, the word *firewall* generated no problems during the experiments in English.

Author Contributions

This work is part of S.G.'s Master's thesis (Góngora 2025), for which L.C. and G.M. served as co-advisors. P.G. helped frame the conceptual aspects of this work.

Acknowledgments

This paper was partially funded by *Agencia Nacional de Investigación e Innovación* (ANII, Uruguay), Grant No. $POS\_NAC\_2022\_1\_173659$; *Comisión Académica de Posgrado* (CAP, Uruguay), Grant No. $BDDX\_2026\_1\#48948494$; and by the project DARK NITE: *Dialogue Agents Relying on Knowledge-Neural hybrids for Interactive Training Environments*, Grant No. $PID2023-146308OB-I00$ (Spanish Ministry of Science and Innovation).

References

Callison-Burch, C.; Tomar, G. S.; Martin, L. J.; Ippolito, D.; Bailis, S.; and Reitter, D. 2022. Dungeons and dragons as a dialog challenge for artificial intelligence. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 9379–9393. Association for Computational Linguistics.

Ferreira, C. 2026. Co-creating narratives: Ludonarrative agency and generative ai in digital games. In *Reshaping the Video Game Landscape With AI and GenAI*. IGI Global Scientific Publishing. 87–122.

Góngora, S.; Chiruzzo, L.; Méndez, G.; and Gervás, P. 2023. Skill Check: Some considerations on the evaluation of gamemastering models for role-playing games. In *International Conference on Games and Learning Alliance*, 277–288. Springer.

Góngora, S.; Chiruzzo, L.; Méndez, G.; and Gervás, P. 2024. PAYADOR: A minimalist approach to grounding language models on structured data for interactive storytelling and role-playing games. In *Proceedings of The 15th International Conference on Computational Creativity*.

Góngora, S. 2025. Approaches to interactive and improvisational storytelling. Master's thesis, Universidad de la República (Uruguay). Facultad de Ingeniería.

Hausknecht, M.; Ammanabrolu, P.; Côté, M.-A.; and Yuan, X. 2020. Interactive fiction games: A colossal adventure. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 7903–7910.

Katifori, A.; Petousi, D.; Sakellariadis, P.; Roussou, M.; and Ioannidis, Y. 2022. Tabletop role playing games and creativity: The game master perspective. In *Proceedings of the 17th International Conference on the Foundations of Digital Games*, FDG '22. Association for Computing Machinery.

Martin, L. J.; Harrison, B.; and Riedl, M. O. 2016. Improvisational computational storytelling in open worlds. In *Interactive Storytelling: 9th International Conference on Interactive Digital Storytelling, ICIDS 2016, Los Angeles, CA, USA, November 15–18, 2016, Proceedings 9*, 73–84. Springer.

Mäyrä, F. 2017. Dialogue and interaction in role-playing games. *Dialogue across Media* 28:271.

Reed, A. A. 2023. *50 Years of Text Games: From Oregon Trail to AI Dungeon*. Changeful Tales Press.

Riedl, M. O., and Bulitko, V. 2012. Interactive narrative: An intelligent systems approach. *AI Magazine* 34(1):67.

Song, J.; Zhu, A.; and Callison-Burch, C. 2024. You have thirteen hours in which to solve the labyrinth: Enhancing ai game masters with function calling. In *The 4th Wordplay: When Language Meets Games Workshop*.

Teleki, M.; Bengali, V.; Dong, X.; Janjur, S. T.; Liu, H.; Liu, T.; Wang, C.; Liu, T.; Zhang, Y.; Shipman, F.; and Caverlee, J. 2025. A survey on LLMs for story generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, 13954–13966. Association for Computational Linguistics.

Tian, Y.; Huang, T.; Liu, M.; Jiang, D.; Spangher, A.; Chen, M.; May, J.; and Peng, N. 2024. Are large language models capable of generating human-level narratives? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 17659–17681. Association for Computational Linguistics.

Trichopoulos, G.; Alexandridis, G.; and Caridakis, G. 2023. A survey on computational and emergent digital storytelling. *Heritage* 6(2):1227–1263.

Triyason, T. 2023. Exploring the potential of chatgpt as a dungeon master in dungeons & dragons tabletop game. In *Proceedings of the 13th International Conference on Advances in Information Technology*, IAIT '23. Association for Computing Machinery.

Tychsen, A.; Hitchens, M.; Brolund, T.; and Kavakli, M. 2005. The game master. In *Proceedings of the Second Australasian Conference on Interactive Entertainment*, IE '05, 215–222. Creativity & Cognition Studios Press.

Vaucher, M.; Silveira, S.; Góngora, S.; and Chiruzzo, L. 2026. IVIE: A neuro-symbolic approach to incremental and validated generation of interactive fiction worlds. In *Proceedings of The 17th International Conference on Computational Creativity*.

Wang, R.; Todd, G.; Xiao, Z.; Yuan, X.; Côté, M.-A.; Clark, P.; and Jansen, P. 2024. Can language models serve as text-based world simulators? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 1–17. Association for Computational Linguistics.

Xie, K.; Yang, I.; Gunerli, J.; and Riedl, M. 2025. Making large language models into world models with precondition and effect knowledge. In *Proceedings of the 31st International Conference on Computational Linguistics*, 7532–7545. Association for Computational Linguistics.

You, X.; Taveekitworachai, P.; Chen, S.; Can Gursesli, M.; Li, X.; Xia, Y.; and Thawonmas, R. 2024. Dungeons, dragons, and emotions: A preliminary study of player sentiment in LLM-driven TTRPGs. In *Proceedings of the 19th International Conference on the Foundations of Digital Games*, FDG '24. Association for Computing Machinery.

Appendices

World-state components

Table 1 shows the four classes present in our system, strongly based on the PAYADOR source code (`Locations`, `Characters`, `Objects` and `Puzzles`). All these components are tracked by a `World` class, representing the *world state*.

Item	**Location**
Name: **String** *Descriptions*: List [**String**] *Gettable*: **Boolean**	*Name*: **String** *Descriptions*: List [**String**] *Items*: List [**Item**] *Connecting locations*: List [**Location**] *Blocked locations*: List [**Location**, *obstacle*]
Character	**Puzzle**
Name: **String** *Descriptions*: List [**String**] *Location*: **Location** *Inventory*: List [**Item**]	*Name*: **String** *Descriptions*: List [**String**] *Problem*: **String** *Answer*: **String**

Table 1: The attributes of each `World` component in our representation. The *obstacle* of a blocked location can be any other component, but it is particularly thought to be an `Item` or a `Puzzle`.

Examples of rendered world-states

Starting scene — Scenario A (English)

```
The player is in <Art studio>
From <Art studio> the player can access: <Kitchen>
From <Art studio> there are blocked passages to:
↪  None
The player has the following objects in the
↪  inventory: None
The player can see the following objects: <A grey
↪  hammer>, <A green hammer>
The player can see the following characters: <Laura>

Here is a description of each component.
<Art studio>: This is the player's location. This is
↪  the art studio that Emma's mom has in the house.
Characters:
- <Player>: The player is acting as <Emma>. A
↪  teenager of average height. She is looking for
↪  her pet 'Hojita'.
- <Laura>: A woman in her 40s. She is Emma's mom.
↪  She is an artist, and loves oil painting. This
↪  character has the following items: <Key>
Objects:
- <A grey hammer>: A big grey hammer that can be
↪  used to break things. It is so heavy...
- <A green hammer>: A small green hammer. It is just
↪  a toy and you cannot break anything with it
- <Key>: A key to open a lock. It is golden. There
↪  is a strange coat of arms engraved on it
```

Starting scene — Scenario B (English)

```
The player is in <Clearing in the woods>
From <Clearing in the woods> the player can access:
↪  None
From <Clearing in the woods> there are blocked
↪  passages to: <Silent zone> blocked by <Firewall>
The player has the following objects in the
↪  inventory: <Guitar>
The player can see the following objects:
↪  <Writings>, <Pond>
The player can see the following characters: None

Here is a description of each component.
<Clearing in the woods>: This is the player's
↪  location. A clearing in a eucalyptus forest near
↪  the Uruguay River. You can hear the sound of the
↪  animals that live in the trees of this forest..
Characters:
- <Player>: The player is acting as <Venancio>. A
↪  Uruguayan gaucho in his 40s. He belongs to the
↪  Artigas army. He has the magical power to summon
↪  a giant wave of water with which he can put out
↪  fires or moisten the ground..
Objects:
- <Writings>: There is something written on the
↪  wall.. It says 'You have to trust in the powers
↪  that have been given to you.'
- <Pond>: A pond full of crystal clear water. The
↪  water is so clear that it works like a mirror
- <Guitar>: A classic guitar with 6 strings. It
↪  sounds great
- <Firewall>: The flames are very hot. It's 3 metres
↪  high. It is impossible to cross them, neither
↪  walking, nor running, nor jumping.
```

World-update prompt

World-update prompt (English)

```
system_msg = f"""You are a storyteller. You are
↪  managing a fictional world, and the player can
↪  interact with it. Following a specific format,
↪  that I will specify below, your task is to find
↪  the changes in the world after the actions in
↪  the player input. Specifically, you will have to
↪  find what objects were moved, which previously
↪  blocked passages are now unblocked, and if the
↪  player moved to a new place.

Here are some clarifications:
(A) Pay attention  to the description of the
↪  components and their capabilities.
(B) If a passage is blocked, then the player must
↪  unblock it before being able to reach the place.
↪  Even if the player tells you that he is going to
↪  access the locked location, you have to be sure
↪  that he is complying with what you asked to
↪  allow him to unlock the access, for example by
↪  using a key or solving a puzzle.
(C) Do not assume that the player input always makes
↪  sense; maybe those actions try to do something
↪  that the world does not allow.
(D) Follow always the following format with the
↪  three categories, using "None" in each case if
↪  there are no changes and repeat the category for
↪  each case:
- Moved object: <object> now is in <new_location>
```

```
- Blocked passages now available:
↪  <now_reachable_location>
- Your location changed: <new_location>
(E) Finally, you can narrate the changes you've
↪  detected in the world state (without moving the
↪  story forward and without making up details not
↪  included in the world state!) using the format:
↪  #your final message#
(F) In the narration section that you add at the
↪  end, between # symbols, you can also answer
↪  questions that the player asks in their input,
↪  about the objects or characters they can see, or
↪  the place they are in.

Here I give you some examples (in parentheses, a
↪  clarification about what the player might have
↪  tried to do) for the asked format, as described
↪  in items (D) and (E):

Example 1 (The player took the axe and put it in the
↪  inventory)
- Moved object: <axe> now is in <Inventory>
- Blocked passages now available: None
- Your location changed: None
#You put the axe in your bag#

Example 2 (The player unblocks the passage to the
↪  basement)
- Moved object: None
- Blocked passages now available: <Basement>
- Your location changed: None
# The basement is now reachable #

Example 3 (The player now is in the garden)
- Moved object: None
- Blocked passages now available: None
- Your location changed: <Garden>
# You enter the garden #

Example 4 (The player puts objects in the bag and
↪  leaves the axe on the floor)
- Moved object: <banana> now is in <Inventory>,
↪  <bottle> now is in <Inventory>,  <axe> now is in
↪  <Main Hall>
- Blocked passages now available: None
- Your location changed: None
# You put the banana and the bottle in your bag. The
↪  axe lies on the floor of the Main hall #

Example 5 (The player puts objects in the bag and
↪  leaves the axe on the floor and unblocks the
↪  passage to the Small room)
- Moved object: <banana> now is in <Inventory>,
↪  <bottle> now is in <Inventory>,  <axe> now is in
↪  <Main Hall>
- Blocked passages now available: <Small room>
- Your location changed: None
# You put the banana and the bottle in your bag. The
↪  axe lies on the floor of the Main hall. Now you
↪  can reach the Small room. #

Example 6 (The player puts objects in the bag and
↪  leaves the axe on the floor, unblocks the
↪  passage and goes to the Small room)
- Moved object: <banana> now is in <Inventory>,
↪  <bottle> now is in <Inventory>,  <axe> now is in
↪  <Main Hall>
- Blocked passages now available: <Small room>
- Your location changed:  <Small room>
# You put the banana and the bottle in your bag. The
↪  axe lies on the floor of the Main hall. The
↪  Small room is now unblocked, and you moved
↪  there. #

Example 7 (The player puts the pencil in the bag and
↪  gives the book to John)
- Moved object: <book> now is in <John>,  <pencil>
↪  now is in <Inventory>
- Blocked passages now available: None
- Your location changed:  None
# John now has the book. You put the pencil in your
↪  bag #

Example 8 (The player gives the computer to Susan)
- Moved object: <computer> now is in <Susan>
- Blocked passages now available: None
- Your location changed:  None
# Susan put the computer in her bag #

Example 9 (The player does something that has not
↪  the expected outcome)
- Moved object: None
- Blocked passages now available: None
- Your location changed:  None
# Nothing happened... #

Example 10 (The player asks a question)
- Moved object: None
- Blocked passages now available: None
- Your location changed:  None
# Answer to the player's question #"""

user_msg = f"""Give the changes in the world
↪  following the specified format, after this
↪  player input "{input}" on this world state:
↪  {world_state}"""
```

Scenario results

In this appendix we include tables with the results of the manual analysis of the playthroughs. These tables (Table 2 and Table 3) show LLM-related errors alongside errors stemming from symbolic world-modeling limitations.

Scenario illustrations

In this appendix we include illustrations to help visualizing the pre-made scenarios we designed to test *Transformations*. These figures (Figure3 and Figure4) are for illustrative purposes only; players interacted with the environment via a text-only interface.

A real example using Gemini

Figure 5 shows a real example using Gemini 1.5 Flash, with step-by-step annotations of human utterances and LLM outputs (both GM utterances and suggested *transformations*).

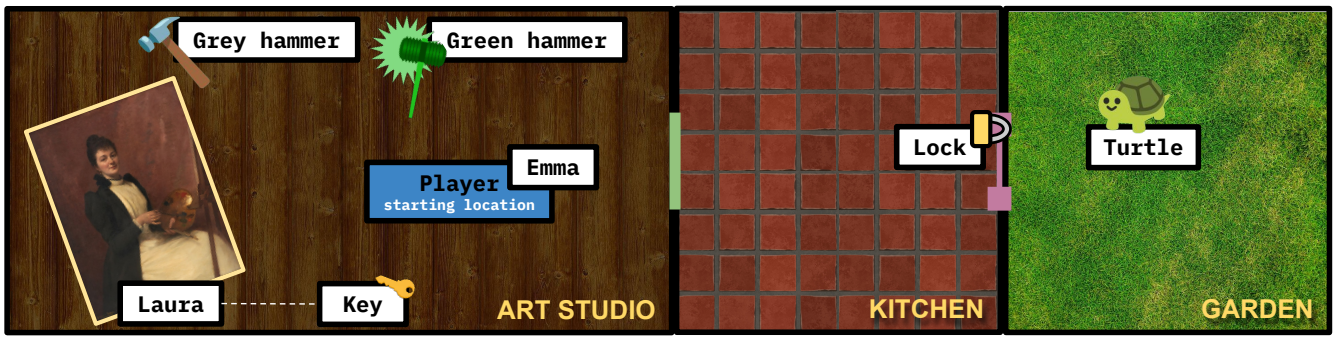

Figure 3: A graphical representation of Scenario A. Ten `world` components are represented: 2 `characters` (*Laura*, an NPC, and *Emma*, the player character), 5 `items` (*Grey hammer*, *Green hammer*, *Key*, *Lock* and *Turtle*, and 3 `locations` (*Art Studio*, *Kitchen* and *Garden*).

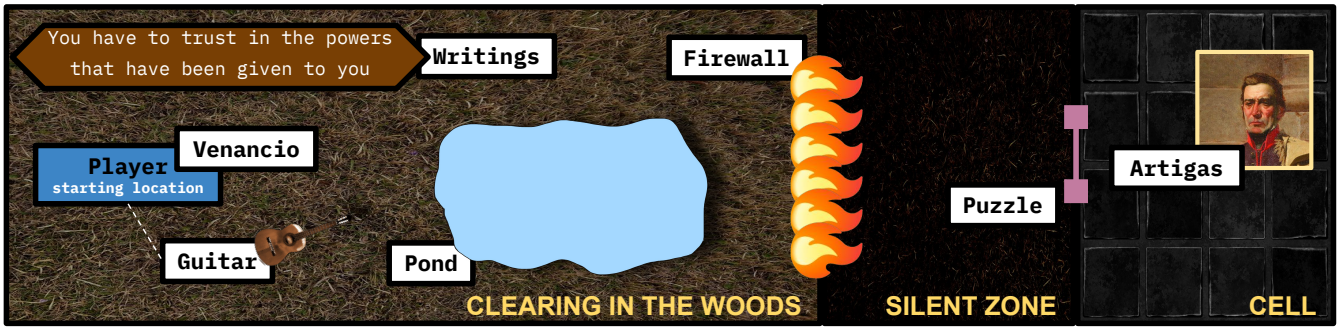

Figure 4: A graphical representation of Scenario B. Ten `world` components are represented: 2 `characters` (*Artigas*, an NPC, and *Venancio*, the player), 4 `items` (*Guitar*, *Pond*, *Writings*, and *Firewall*), 3 `locations` (*Clearing in the woods*, *Silent zone* and *Cell*), and a `puzzle` (*Puzzle*)

Scenario A	LLM Errors			World Modeling Errors	
	MI-t	PM-t	UL-t	Planning	Memory
TesterA (Gemini)	0	0	0	1	0
TesterB (Gemini)	0	0	0	1	0
TesterC (Llama)	3	1	0	0	0
TesterD (Llama)	4	1	0	2	0
Total (English)	7	2	0	4	0
TesterE (Gemini)	3	0	0	2	1
TesterF (Gemini)	1	0	0	0	0
TesterG (Llama)	4	0	1	1	0
TesterH (Llama)	0	0	0	0	0
Total (Spanish)	8	0	1	3	1

Table 2: Number of errors by category for each of the eight testers during the playthroughs of Scenario A. Testers A-D played in English, while Testers E-H played in Spanish. The LLM used is indicated for each case, next to the tester's id.

Scenario B	LLM Errors			World Modeling Errors	
	MI-t	PM-t	UL-t	Planning	Memory
TesterA (Gemini)	0	0	1	0	0
TesterB (Gemini)	0	0	1	0	0
TesterC (Llama)	0	0	1	0	0
TesterD (Llama)	0	0	0	0	0
Total (English)	0	0	3	0	0
TesterE (Gemini)	1	1	2	0	0
TesterF (Gemini)	0	2	1	0	0
TesterG (Llama)	0	1	0	0	0
TesterH (Llama)	0	1	0	0	0
Total (Spanish)	1	5	3	0	0

Table 3: Number of errors by category for each of the eight testers during the playthroughs of Scenario B. Testers A-D played in English, while Testers E-H played in Spanish. The LLM used is indicated for each case, next to the tester's id.

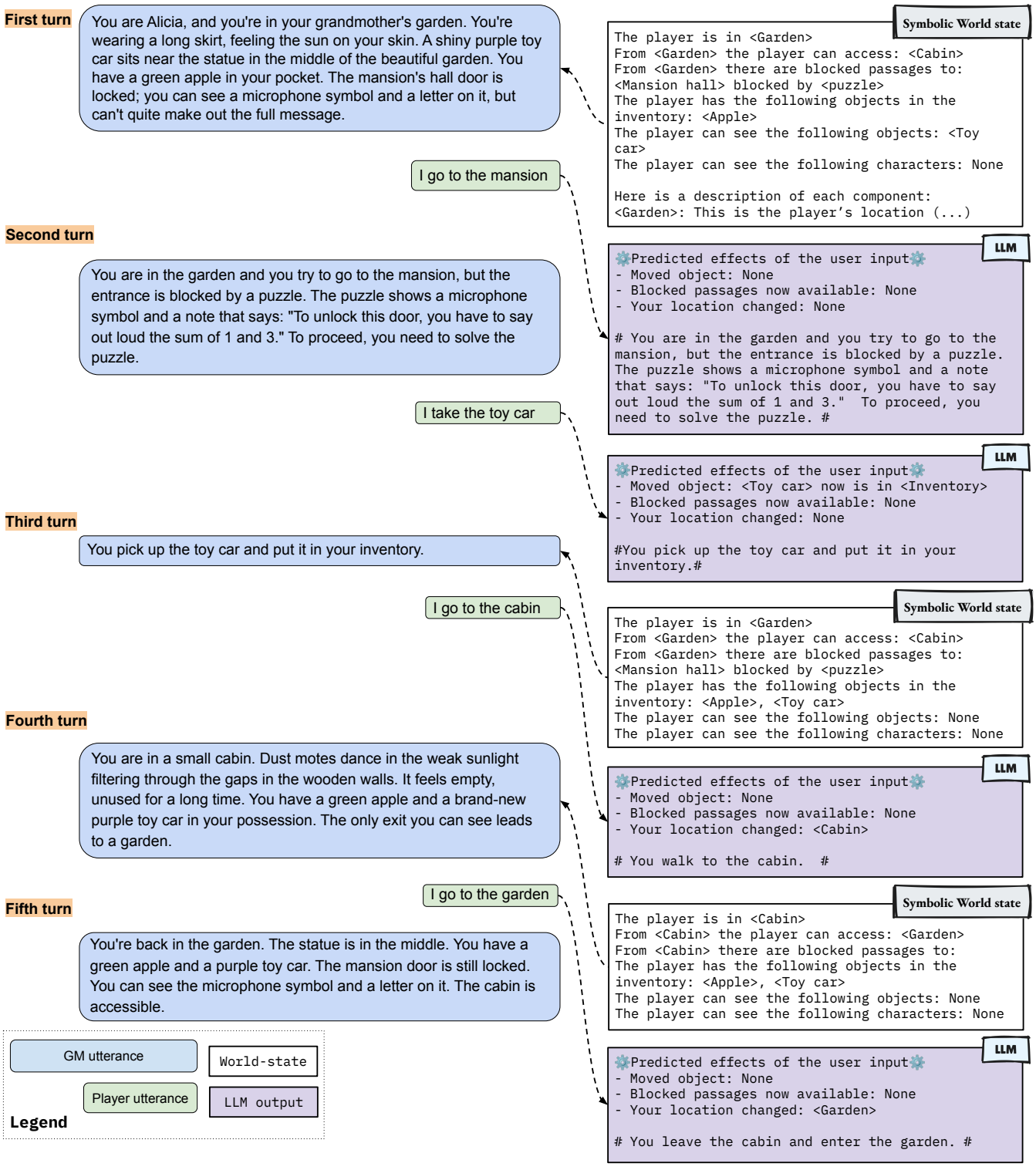

Figure 5: A five-turn gameplay example using Gemini 1.5 Flash. The left panel displays the dialogue between the automated Game Master and the human player, while the right panel shows the symbolic world states (for representative turns) and the LLM-suggested *transformations* (for all turns). A legend is provided in the bottom left-hand corner.